»Лекционен курс

»ООП1 (Java)

Масиви ›

Мотивация

Въвеждащ пример

- Да предположим, че искаме да изчислим средната температура за седемте дни в седмицата.
- Може да използваме следния код:

```
Scanner keyboard = new Scanner(System.in);
System.out.println("Enter 7 temperatures:");
double sum = 0;
for (int count = 0; count < 7; count++){
  double next = keyboard.nextDouble();
  sum = sum + next;
}
double average = sum / 7;
```

Мотивация

Какво трябва да направим?

- Това работи добре, ако всичко, което искаме да знаем, е средната стойност
- Да приемем, че искаме също да знаем кои температури са над, съответно под средните

```
Scanner keyboard = new Scanner(System.in);
System.out.println("Enter 7 temperatures:");
double sum = 0;
for (int count = 0; count < 7; count++){
  double next = keyboard.nextDouble();
  sum = sum + next;
}
double average = sum / 7;
```

Мотивация

```
Scanner keyboard = new Scanner(System.in);
System.out.println("Enter 7 temperatures:");
double sum = 0;
for (int count = 0; count < 7; count++){
  double next = keyboard.nextDouble();
  sum = sum + next;
}
double average = sum / 7;
```

Сега имаме проблем

- *За да изчислим средната стойност, трябва да прочетем седемте температури и да изчислим средната стойност, преди да сравним всяка температура с нея*
- *По този начин, за да можем да сравним всяка температура със средната, трябва да запомним седемте температури*

Мотивация

- Какво можем да направим?
  - Очевидният отговор е да се използват седем променливи от тип double
  - Това е малко неудобно, защото трябва да декларираме седем променливи
- В други ситуации проблемът може да бъде още по-лош
  - Да си представим, че искаме това за всеки ден от годината,
  - Задаването на 365 декларации на променливи би било абсурдно
- Масивите ни предоставят елегантен начин да декларираме колекция от свързани променливи

Определение и създаване

Определение, обща характеристика

- Масив: колекция от променливи от един и същ тип, реферирани от общо име
  - Той е нещо като списък с променливи, но се справя с именуването на променливите по приятен, компактен начин
- В Java масивите могат да едномерни или многомерни
  - Едномерните са най-често използвани
- Масивите се използват за различни цели
  - Предлагат удобни средства за групиране на съответните променливи
- Основното предимство на един масив е, че той организира данните по такъв начин, че лесно да бъдат манипулирани

Създаване на масиви

Масивите в Java се създават като обекти!

- В Java масивът е специален вид обект
- Често е по-полезно да се мисли за масив като колекция от променливи от същия тип
- Напр., масив, състоящ се от колекция от седем променливи от тип double, може да бъде създаден, както следва:

```
double[] temperature = new double[7];
```

- Това е като да декларираме следните (не съвсем обичайно именувани) седем променливи от тип double:

```
temperature[0], temperature[1], temperature[2], temperature[3],
temperature[4], temperature[5], temperature[6]
```

- Променливи като temperature[0], които имат целочислен израз в квадратни скоби, се наричат индексирани променливи (елементи на масив или просто елементи).
- Целочисленият израз в квадратните скоби се нарича индекс
- В Java номерирането започва от 0.

Създаване на масиви

Декларация:

```
double[] temperatures;
String[] weekDays;
```

Възможно ли е също?

```
double temperatures[];
String weekDays[];
```

temperatures:

Създаване:

```
temperatures = new double [20];
weekDays = new String [7];
```

От тук нататък дължината е позната

temperatures:

Създаване на масиви

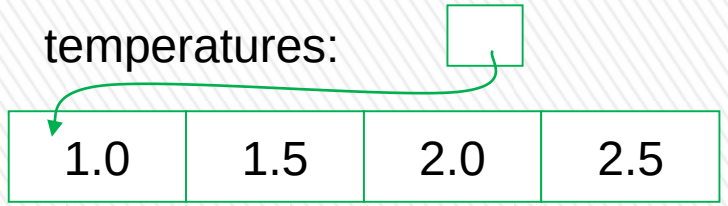
Декларация и създаване обединени

```
double [] temperatures
   = new double [20];
```

Още при декларацията броят на елементите е познат

Създаване посредством начална стойност

```
double [] temperatures
     = {1.0, 1.5, 2.0, 2.5};
```



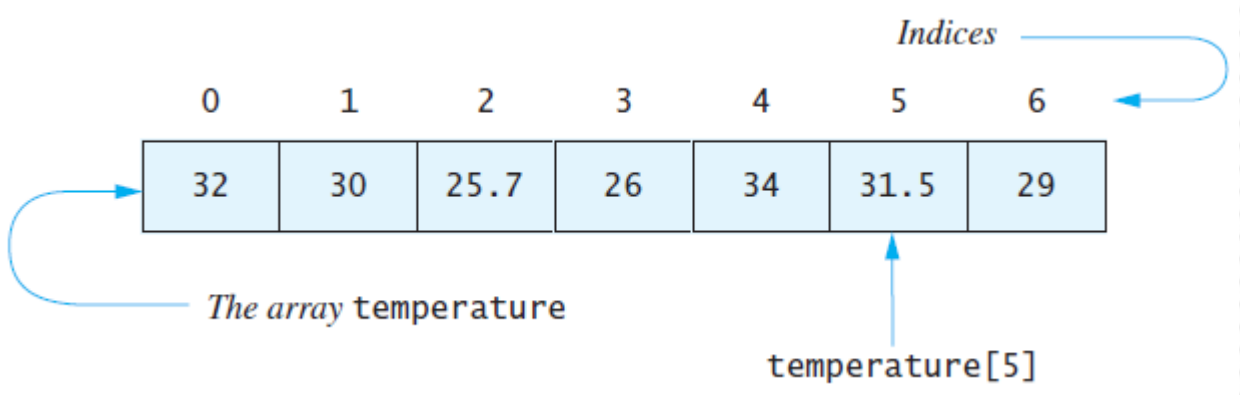
Работа с масиви

Достъп до масиви

- Всяка от тези седем променливи може да се използва точно както всяка друга променлива от тип double.
- Напр., в Java са разрешени всички тези оператори:

```
temperature[3] = 32;
temperature[6] = temperature[3] + 5;
System.out.println(temperature[6]);
```

- Ако приемем, че тези индексирани променливи са групирани в един общ елемент, ще ги наречем масив.
- Така че можем да се позовем на масива, наречен temperature, без да използваме квадратни скоби.



Пример

Какъв резултат?

```
import java.util.Scanner;
public class ArrayOfTemperatures2 {
    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);
        System.out.println("Колко температури искате?");
        int size = keyboard.nextInt( );
        double[] temperature = new double[size];
        System.out.println("Въведете " + temperature.length + " температурите:");
        double sum = 0;
        for (int index = 0; index < temperature.lengt
            temperature[index] = keyboard.nextDouble(
            sum = sum + temperature[index];
        }
        double average = sum / temperature.length;
        System.out.println("Средната температура е "
        System.out.println("Температурите са");
        for (int index = 0; index < temperature.lengt
            if (temperature[index] < average)
                System.out.println(temperature[index]
            else if (temperature[index] > average)
                System.out.println(temperature[index]
            else //temperature[index] == average
                System.out.println(temperature[index]
        }
        System.out.println("Приятна седмица.");
    }
}
```

```
Колко температури искате?
3
Въведете 3 температурите:
23,5
26,7
31,2
Средната температура е 27.133333333333336
Температурите са
23.5 под средната
26.7 под средната
31.2 над средната
Приятна седмица.

Process finished with exit code 0
```

Инициализация на масиви

- Масивът може да бъде инициализиран по времето, когато е деклариран.
- Пример: еквивалентни декларации

```
double[] reading = {3.3, 15.8, 9.7};
```

```
double[] reading = new double[3];
reading[0] = 3.3;
reading[1] = 15.8;
reading[2] = 9.7;
```

- Ако не инициализираме елементите на масив, те могат автоматично да бъдат инициализирани по подразбиране за основния тип.
- Например, ако не инициализирате масив от цели числа, всеки елемент от масива ще бъде инициализиран с 0.
- Обикновено е по-ясно да направим своя собствена явна инициализация.
- Можем да инициализираме масив както горе или чрез присвояване на стойности в цикъл

```
int[] count = new int[100];
for (int i = 0; i < 100; i++)
    count[i] = 0;
```

Пример

```
import java.util.*;
public class ArrayNew {
   static Random rand = new Random();
   static int pRand(int mod) {
       return Math.abs(rand.nextInt()) % mod + 1;
   }

   public static void main(String[ ] args) {
      int[ ] a;
      a = new int[pRand(20)];
      System.out.println("length of a = " + a.length);
      for(int i = 0; i < a.length; i++)
         System.out.println("a[" + i + "] = " + a[i]);
   }
}
```

Броят на елементите в масива не е предварително (по време на програмиране) известен

Пример

Създава нов генератор на случайни числа

```
import java.util.*;
public class ArrayNew {
   static Random rand = new Random();
   static int pRand(int mod) {
       return Math.abs(rand.nextInt()) % mod + 1;
   }

   public static void main(String[ ] args) {
      int[ ] a;
      a = new int[pRand(20)];
      System.out.println("length of a = " + a.length);
      for(int i = 0; i < a.length; i++)
         System.out.println("a[" + i + "] = " + a[i]);
   }
}
```

Генерира случайна целочислена стойност и връща нейната абсолютна стойност в определен интервал

Пример

```
import java.util.*;
public class ArrayNew {
   static Random rand = new Random();
   static int pRand(int mod) {
       return Math.abs(rand.nextInt()) % mod + 1;
   }

   public static void main(String[ ] args) {
      int[ ] a;
      a = new int[pRand(20)];
      System.out.println("length of a = " + a.length);
      for(int i = 0; i < a.length; i++)
         System.out.println("a[" + i + "] = " + a[i]);
   }
}
```

Не е задължителен операторът new

Пример

```
import java.util.*;
public class ArrayNew {
   static Random rand = new Random();
   static int pRand(int mod) {
       return Math.abs(rand.nextInt()) % mod + 1;
   }

   public static void main(String[ ] args) {
      int[ ] a;
      a = int[pRand(20)];
      System.out.println("length of a = " + a.length);
      for(int i = 0; i < a.length; i++)
         System.out.println("a[" + i + "] = " + a[i]);
   }
}
```

Пример

Какъв резултат?

```
import java.util.*
public class Array
   static Random r
   static int pRan
       return Math
   }
```

```
length of a = 10
a[0] = 0
a[1] = 0
a[2] = 0
a[3] = 0
a[4] = 0
a[5] = 0
a[6] = 0
a[7] = 0
a[8] = 0
```

```
length of a = 9
a[0] = 0
a[1] = 0
a[2] = 0
a[3] = 0
a[4] = 0
a[5] = 0
a[6] = 0
a[7] = 0
a[8] = 0

Process finished with exit code 0
```

```
length of a = 18
a[0] = 0
a[1] = 0
a[2] = 0
a[3] = 0
a[4] = 0
a[5] = 0
a[6] = 0
a[7] = 0
a[8] = 0
a[9] = 0
a[10] = 0
a[11] = 0
a[12] = 0
a[13] = 0
a[14] = 0
a[15] = 0
a[16] = 0
a[17] = 0

Process finished with exit code 0
```

Пример – масиви от обекти

Коментар?

```
import java.util.*;
public class ArrayClassObj {
   static Random rand = new Random();
   static int pRand(int mod) {
      return Math.abs(rand.nextInt()) % mod + 1;
   }
   public static void main(String[] args) {
      Integer[] a = new Integer[pRand(20)];
      System.out.println("length of a = " + a.length);
      for(int i = 0; i < a.length; i++) {
          a[i] = new Integer(pRand(500));
          System.out.println("a[" + i + "] = " + a[i]);
       }
   }
}
```

- Класът Integer обвива стойност на int на примитивен тип в обект
- Обект от тип Integer съдържа едно поле, чийто тип е int

Класове-обвивки

Primitive Data Type	Wrapper Class
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
boolean	Boolean
char	Character

Пример

```
import java.util.*;
public class ArrayClassObj {
   static Random rand = new Random();
   static int pRand(int mod) {
      return Math.abs(rand.nextInt()) % mod + 1;
   }
   public static void main(String[] args) {
      Integer[] a = new Integer[pRand(20)];
      System.out.println("length of a = " + a.length);
      for(int i = 0; i < a.length; i++) {
          a[i] = new Integer(pRand(500));
          System.out.println("a[" + i + "] = " + a[i]);
       }
   }
}
```

Когато работим с масиви от обекти, тогава задължително използваме new

Пример

Какъв резултат?

```
import java.util.*;
public class ArrayClassObj {
    static Random rand = new
    static int pRand(int mod)
        return Math.abs(rand.n
    }
    public static void main(String[] args) {
        Integer[] a = new Integer[pRand(20)];
        System.out.println("length of a = " + a.
        for(int i = 0; i < a.length; i++) {
            a[i] = new Integer(pRand(500));
            System.out.println("a[" + i + "] = "
         }
    }
}
```

```
length of a = 5
a[0] = 49
a[1] = 359
a[2] = 264
a[3] = 237
a[4] = 194

Process finished with exit code 0
```

```
length of a = 9
a[0] = 133
a[1] = 50
a[2] = 60
a[3] = 9
a[4] = 484
a[5] = 187
a[6] = 189
a[7] = 159
a[8] = 434

Process finished with exit code 0
```

Присвояване на масиви

Възможно?

```
public class Arrays {
    public static void main(String[] args) {
        int[ ] a1 = { 1, 2, 3, 4, 5 };
        int a2[ ];

        a2 = a1;

        for(int i = 0; i < a2.length; i++ )
            a2[i]++;

        for(int i = 0; i < a1.length; i++ )
            System.out.println("a1[" + i + "] = " + a1[i]);
    }
}
```

Присвояване на масиви

Какъв резултат и коментар?

```
public class Arrays {
    public static void main(String[] args) {
        int[ ] a1 = { 1, 2, 3, 4, 5 };
        int a2[ ];

        a2 = a1;

        for(int i = 0; i < a2.length; i++ )
            a2[i]++;

        for(int i = 0; i < a1.length; i++ )
            System.out.println("a1[" + i + "] = " + a1[i]);
    }

}
```

```
a1[0] = 2
a1[1] = 3
a1[2] = 4
a1[3] = 5
a1[4] = 6

Process finished with exit code 0
```

Програмни параметри

Програмни параметри

Коментар

```
class Temperature {
   // Convert temperature
   // from Fahrenheit to Centigrade

   public static void main (String[] args) {
        double tempFahr; // Fahrenheit
        double tempCels; // Celsius
        System.out.print("Temperature (deg F): ");
        tempFahr = Keyboard.readDouble();
        tempCels = (5.0 * (tempFahr - 32.0)) / 9.0;
        System.out.print(tempFahr);
        System.out.print(" deg F is ");
        System.out.print(tempCels);
        System.out.println(" deg C");
   }
}
```

```
% javac Temperature.java
% java Temperature

Temperature (deg F): 10
10 deg F is -12.222222222222221 deg C
```

Програмни параметри

```
class select {
   public static void main (String[] args) {
      ...
   }
}
```

Формален параметър от тип Typ 'Array' от 'String'

От къде аргументът?

Програмни параметри

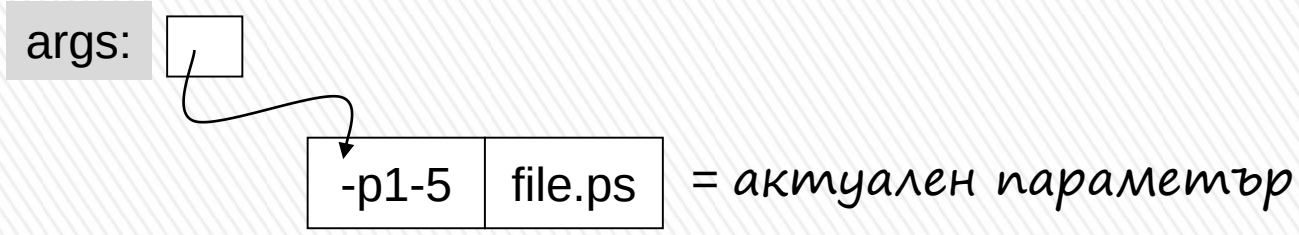
```
class select {
   public static void main (String[] args) {
      ...
   }
}
```

Формален параметър от тип Typ 'Array' от 'String'

От къде аргументът?

```
% java select -p1-5 file.ps
```

от командния ред



= *актуален параметър*

Използване на програмни параметри

```
% java temp -C -F,K
% java temp -K -C,K
```

Програми, управлявани от предавани по времето на извикването параметри

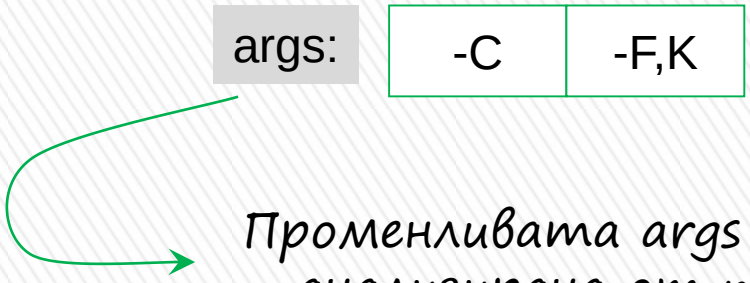
Достъпен ли args в main метода?

Използване на програмни параметри

```
% java temp -C -F,K
% java temp -K -C,K
```

Програми, управлявани от предавани по времето на извикването параметри

Достъпен ли args в main метода?



Програмни параметри

Какво прави програмата?

```
public class Echo {
   public static void main(String args[]) {
      for (int i = 0; i < args.length; i++)
         System.out.print(args[i] + "\n");
   }
}
```

Програмни параметри

Какво прави програмата?

```
public class Echo {
   public static void main(String args[]) {
      for (int i = 0; i < args.length; i++)
         System.out.print(args[i] + "\n");
   }
}
```

```
% java Echo -p1-5,6 f1.ps f2.ps
-p1-5,6
f1.ps
f2.ps
```

args:	-p1-5,6	f1.ps	f2.ps

3 програмни параметри: 1 актуален параметър от тип Array с дължина 3

Exo

Exo в обратен ред

Стара версия:

```
% java Echo -p1-5,6 f1.ps f2.ps
-p1-5,6
f1.ps
f2.ps
```

```
public static void main(String args[]) {
   for (int i=0; i < args.length; i++)
        System.out.print(args[i] + "\n");
}
```

Нова версия:

```
% java Rev -p1-5,6 f1.ps f2.ps
f2.ps f1.ps -p1-5,6
```

```
public static void main(String args[]) {
   for (int i=args.length-1; i >= 0; i--)
        System.out.print(args[i] + " ");
}
```

Специални мсиви

Мрежа на Ератостен

Пример: Да се намерят простите числа до определена граница

Дефиниция: в математиката просто число се нарича всяко естествено число, по-голямо от 1, което има точно два естествени делителя – 1 и самото себе си.

Техника: мрежа на Ератостен

Идея: Задраскваме всички многократни на вече разпознатите прости числа

Използване на логически масиви

Логически масиви

Пример: Мрежа на Ератостен

```
boolean[] net;

// Position i corresponds number i
// net[i] = true <-> i is prim number
```

Начално присвояване

Пример

```
import java.util.Scanner;
public class PrimeSieve {
    public static void main(String[] args) {
        System.out.println("Моля, въведете число:")
        Scanner keyboard = new Scanner(System.in);
        int n = keyboard.nextInt();
        boolean[] isPrime = new boolean[n+1];
        for (int i = 2; i <= n; i++) {
            isPrime[i] = true;
        }
        for (int factor = 2; factor*factor <= n; factor++) {
            if (isPrime[factor]) {
                for (int j = factor; factor*j <= n; j++) {
                    isPrime[factor*j] = false;
                }
            }
        }
        int primes = 0;
        for (int i = 2; i <= n; i++) {
            if (isPrime[i]) primes++;
        }
        System.out.println("Брой на простите числа <= " + n + " е " + primes);
    }
}
```

n	Primes <= n
10	4
100	25
1,000	168
10,000	1,229
100,000	9,592
1,000,000	78,498
10,000,000	664,579
100,000,000	5,761,455
1,000,000,000	50,847,534

Пример

Какъв резултат?

```
Моля, въведете число:
9
Брой на простите числа <= 9 е 4

Process finished with exit code 0
```

```
import java.util.Scanner;
public class PrimeSieve {
    public static void main(String[] args) {
        System.out.println("Моля, въведете число:")
        Scanner keyboard = new Scanner(System.in);
        int n = keyboard.nextInt();
        boolean[] isPrime = new boolean[n+1];
        for (int i = 2; i <= n; i++) {
            isPrime[i] = true;
        }
        for (int factor = 2; factor*factor <= n; factor++) {
            if (isPrime[factor]) {
                for (int j = factor; factor*j <= n; j++) {
                    isPrime[factor*j] = false;
                }
            }
        }
        int primes = 0;
        for (int i = 2; i <= n; i++) {
            if (isPrime[i]) primes++;
        }
        System.out.println("Брой на простите числа <= " + n + " е " + primes);
    }
}
```

Пример

Пример: Управление на дати и месеци

- Имена
- Брой дни

```
public final static String[] MONTH_NAME = { "",
      "January", "February", "March",
      "April", "May", "June",
      "July", "August", "September",
      "October","November", "December"};

public final static int[] DAYS_OF_MONTH = { 0,
      31, 28, 31, 30,
      31, 30, 31, 30,
      30, 31, 30, 31};
```

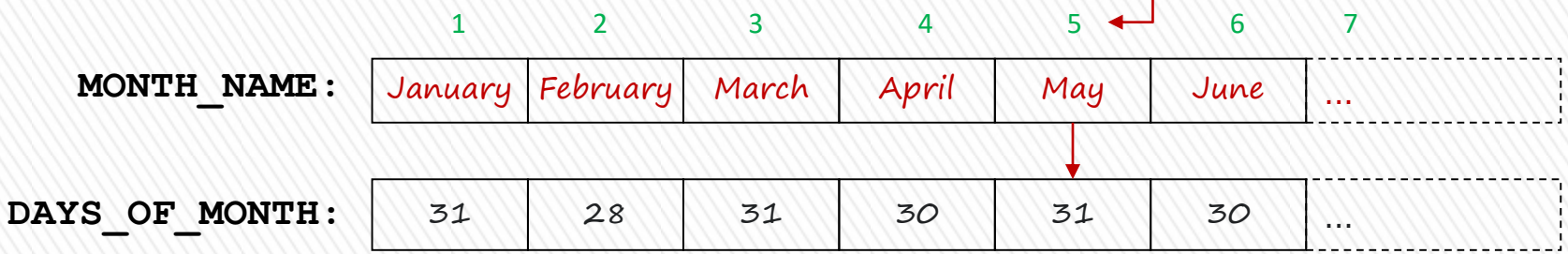
Константни и паралелни масиви

Кореспондиращи двойки

```
public final static String[] MONTH_NAME = { "",
      "January", "February", "March",
      "April", "May", "June",
      "July", "August", "September",
      "October","November", "December"};

public final static int[] DAYS_OF_MONTH = { 0,
      31, 28, 31, 30,
      31, 30, 31, 30,
      30, 31, 30, 31};

final static int MAY = 5;
```



Пример

Какъв резултат?

Month Mai has 31 Days

```
static int M = MAY;

public static void main (String[] args) {
   System.out.println(
      "Month " + MONTH_NAME[M]
      + " has " +
      DAYS_OF_MONTH[M] + " Days");
}
```

Същият индекс

Многомерни масиви

Каква дължина a1?

2

```
public class MultiDimArray {
    public static void main(String[] args) {

     int[ ][ ] a1 = {
                        { 1, 2, 3 },
                        { 4, 5, 6 },
                     };

      for(int i = 0; i < a1.length; i++)
       for(int j = 0; j < a1[i].length; j++)
        System.out.println( "a1["+i+"]["+j+"]="+a1[i][j]);
   }
}
```

Многомерни масиви

Каква дължина a1[i]? 3

```
public class MultiDimArray {
    public static void main(String[] args) {

     int[ ][ ] a1 = {
                        { 1, 2, 3 },
                        { 4, 5, 6 },
                    };

      for(int i = 0; i < a1.length; i++)
       for(int j = 0; j < a1[i].length; j++)
        System.out.println( "a1["+i+"]["+j+"]="+a1[i][j]);
   }
}
```

Многомерни масиви

Какъв резултат?

```
a1[0][0]=1
a1[0][1]=2
a1[0][2]=3
a1[1][0]=4
a1[1][1]=5
a1[1][2]=6

Process finished with exit code 0
```

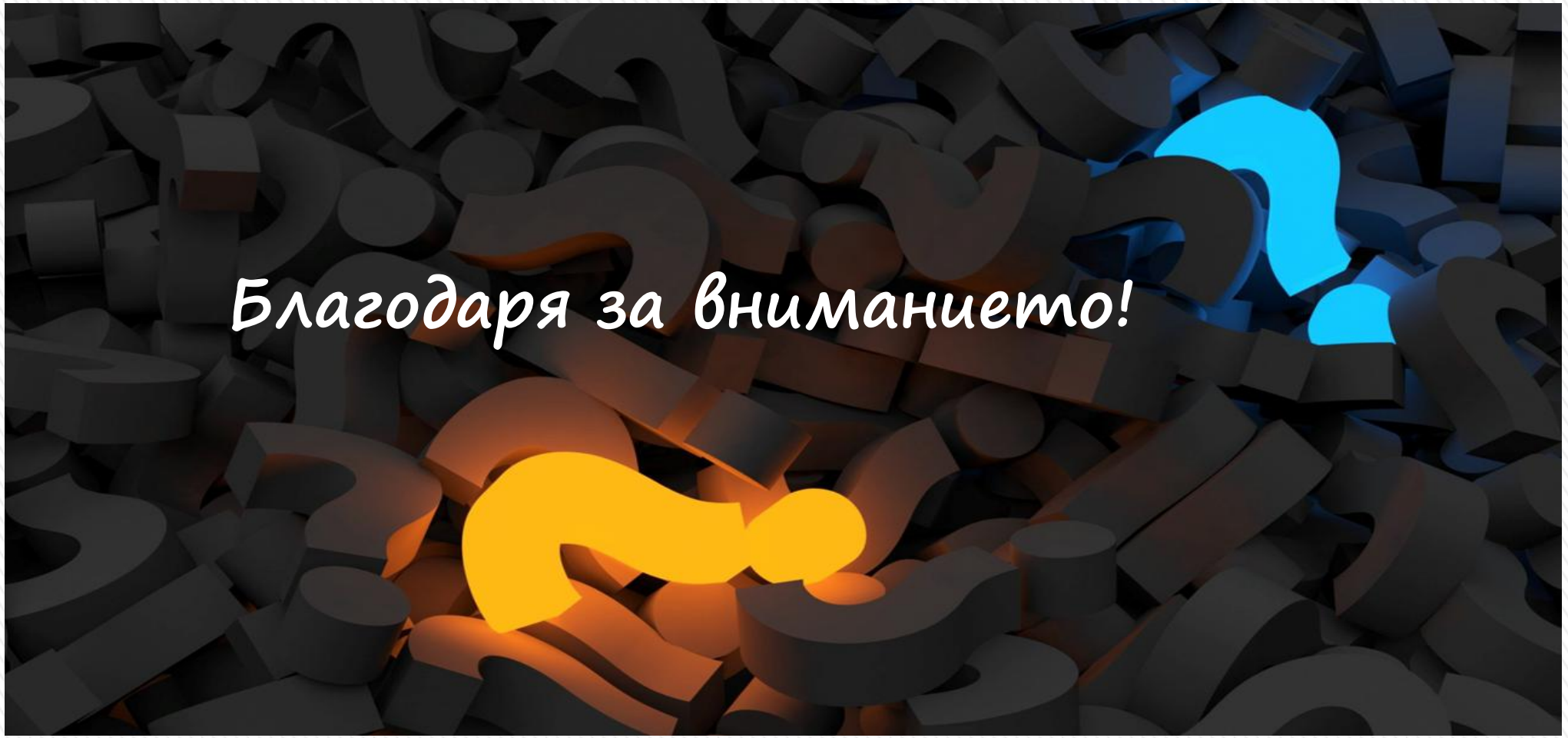
```
public class MultiDimArray {
    public static void main(String[] args) {

     int[ ][ ] a1 = {
                        { 1, 2, 3 },
                        { 4, 5, 6 },
                    };

      for(int i = 0; i < a1.length; i++)
       for(int j = 0; j < a1[i].length; j++)
        System.out.println("a1[" +i+"]["+j+"]="+a1[i][j]);
   }
}
```

Заключение

- Масивите могат да се използват като променливи на обекти
- Методите могат да имат индексирана променлива или цял масив като аргумент и могат да връщат като резултат масиви.
- Накратко, масивите могат да се използват с класове и методи точно както другите обекти.


Благодаря за вниманието!